Durham University

Durham Research Online

Deposited in DRO:

11 December 2015

Version of attached file:

Published Version

Peer-review status of attached file:

Peer-reviewed

Citation for published item:

McWilliam, R. and Schiefer, P. and Purvis, A. (2015) 'Experimental validation of a resilient electronic logic design with autonomous fault discrimination/masking.', Procedia CIRP., 38 . pp. 265-270.

Further information on publisher's website:

http://dx.doi.org/10.1016/j.procir.2015.08.027

Publisher's copyright statement:

© 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license.

Additional information:

Use policy

The full-text may be used and/or reproduced, and given to third parties in any format or medium, without prior permission or charge, for personal research or study, educational, or not-for-profit purposes provided that:

- a full bibliographic reference is made to the original source
- a link is made to the metadata record in DRO
- the full-text is not changed in any way

The full-text must not be sold in any format or medium without the formal permission of the copyright holders.

Please consult the full DRO policy for further details.

Durham University Library, Stockton Road, Durham DH1 3LY, United Kingdom
Tel : +44 (0)191 334 3042 | Fax : +44 (0)191 334 2971
https://dro.dur.ac.uk

Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 38 (2015) 265 – 270

www.elsevier.com/locate/procedia

The Fourth International Conference on Through-life Engineering Services

Experimental validation of a resilient electronic logic design with autonomous fault discrimination/masking

Richard McWilliam*, Philipp Schiefer and Alan Purvis

School of Engineering and Computing Sciences, Durham University, South Road, Durham, DH1 3LE, UK

* Corresponding author. Tel.: +44 (0)191 3342539; fax: +44 (0)191 3342408. *E-mail address:* r.p.mcwilliam@durham.ac.uk

Abstract

This paper presents the experimental validation of a novel fault-tolerant electronic logic design conducted by an automated mixed-signal fault injection procedure. The design under evaluation relies upon a novel redundant design strategy intended to provide fault discrimination and selective fault masking embedded within a functional CMOS NAND gate. The traditional logic layout is modified to include fault detection and reporting at an extremely fine-grained design level with 2x overhead as opposed to the traditional 4x overhead. The fault injection test bench procedure requires automated fault injection, programmable fault load conditions and combined analogue/digital domain verification. The device under test is implemented using discrete n- and p-FETs arranged as a modular test board together with automated fault injection and test lines. The fault response is measured and confirms the predicted intrinsic fault rate of 25% with a successful 100% masking of suck low-faults and precise identification of stuck-high via IDDQ trigger. The test procedure is shown to be extensible towards more complex logic unit designs and for evaluation of multiple simultaneous faults.

© 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Peer-review under responsibility of the Programme Chair of the Fourth International Conference on Through-life Engineering Services.

Keywords: Self-healing technologies; FPGA; hardware-in-loop; fault injection; fault-tolerance

1. Introduction

Future electronics technologies based on nanoscale transistors or printed organic materials will become increasingly vulnerable to process variation and single event upset in harsh environments than present-day technologies. As more and more Commercial Off-the-Shelf (COTS) components are integrated into state of the art high value and mission critical systems, there is also growing concern over the economy of manufacture (i.e. yield) [1] and the in-service robustness of such systems. This is prompting new studies into fault tolerant designs [2] at the extremely fine-grained level that are based in part by the pioneering work of Von Neumann on the theory of massively redundant design [3]. This represents a distinct departure from design strategies that are prevalent today, such as component screening and high-level modular redundancy, and therefore is new design paradigm for future electronics.

Faults that occur in electronics are best categorised into a number of fault domains including: process variation, ageing, current thermal stress, soft errors, hard errors and extreme environmental events that are unforeseen [4]. Within each of these domains, there are many root-causes that illicit faults.

In this paper we present an experimental fault testing platform for verifying novel fault-tolerant design strategies in electronics with both analogue and digital diagnosis. The particular design strategy evaluated in this paper is relevant to low voltage CMOS circuits within ASICs and in power circuits that use discrete transistors such as IGBTs. Here we adopt an abstract model of switching elements based on field-effect transistors (FET) of some kind, and assume that the resource requirements depend upon two factors: the number of switching elements and the number of interconnects. Specific fault conditions are then injected into the hardware and the response recorded.

2212-8271 © 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Peer-review under responsibility of the Programme Chair of the Fourth International Conference on Through-life Engineering Services.
doi:10.1016/j.procir.2015.08.027

Besides the goal of achieving fault tolerance by fine-grained redundancy, a further opportunity exists to consider at the same time fault detection and reporting. Most fine-grained methods are based on the fault masking property i.e., where faults events are made *non-critical* so that they cannot cause an error in the output. Therefore the circuit is able to continue to function correctly in the presence of the fault. An example of this is the quadded logic strategy [5] where gate and intereonnect redundancy guarantee error-free operation for any single fault event. An example is illustrated in Figure 1a. Like most masking strategies, this approach is limited to single faults such as those caused by single event upset (SEU) though in some cases certain combinations of two simultaneous faults can also be tolerated. When SEU occur they cause a momentary charge event to occur within one or more switching elements (i.e., charge is generated in the semiconductor gate of the FET) that may cause a stuck-on or stuck-off fault. Although the fault mechanism is short-lived, it may cause a persistent effect in the circuit that remains until power-off. Hence SEU-induced faults may remain in the circuit for some time, and may potentially accumulate in several locations as is often observed in the SRAM of FPGA chips [6]. If the circuit is configurable (as is the case with FPGA chips) then *scrubbing* is often combined with masking strategies in order to maximise resilience to SEU.

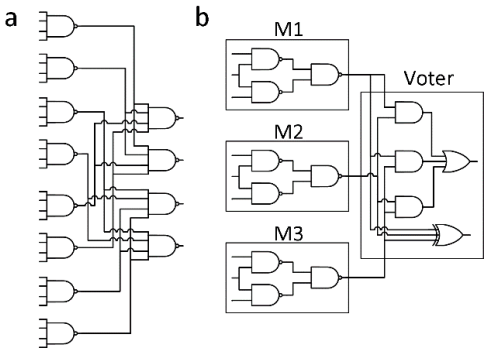

Figure 1 Two examples of fault tolerant design. (a) quadded interleaved logic. (b) TMR with modules M1…M3 and voter logic. The fault detection signal is generated by the XOR gate.

Other fault tolerant strategies apply the concept of triple modular redundancy (TMR) to fine-grained logic in electronics. This has become commonplace in FPGAs for mission-critical situations such as Space exploration where existing designs are augmented by TMR structures using HDL extensions. Several strategies have been developed for FPGA chips that involve various forms of fine-grained modular redundancy, including the capacity for online reconfiguration in response to permanent fault conditions [7,8]. However they are highly confined to the available FPGA resources and architecture while reconfiguration requires a great deal of processing capability or else pre-storage of multiple alternative configurations. The TMR approach has also been applied in multi-processor core design [9]. Despite the presence of built-in checking via the TMR voting logic this is not generally used for reporting at higher system levels.

Error detection and correction (EDC) is also used for *self-checking* by using information redundancy [10]. This approach is most effective for protecting data registers and memory blocks. Early computer systems used combined EDC and modular redundancy [11] but EDC is far more common in state of the art computers. Again however, EDC is usually performed internally with little or no record of fault events. Custom logic, such as the arithmetic logic unit (ALU), are more difficult to modify for fault tolerance since they must be designed with speed and minimal component layout.

At the power electronics scale, discrete IGBTs used in renewable energy power conversion systems are becoming a source of concern. Although their intrinsic reliability is high, modern energy conversion sub-systems may rely upon a large number of such devices, increasing the likelihood of failures occurring. This causes serious disruption for critical energy systems such as offshore wind farms where maintenance and repair is difficult and expensive. Mitigation strategies then become one of further improving component reliability or else introducing efficient fault tolerance with the smallest possible overhead.

As a result of the above context, our fault tolerant design aims to combine the following properties within the confines of custom logic gate design:

- Scalability: the redundancy structure is applicable to low and high level electronic design
- Strategic masking: only stuck-off events are masked
- Strategic fault detection: stuck-on events trigger a built-in fault detection mechanism
- Low overhead: the redundancy overhead is minimised.

Resource overhead represents a balance of cost/benefit where, in this case, the benefits gained are fault selectivity and detection. Goal of this work is to therefore to design and test a hardware prototype that demonstrates the above features of the fault tolerant design.

2. Proposed design

The basis of this design is the NAND logic gate, which is central to electronic ALUs and many other circuits. The reference NAND gate is illustrated in Figure 2a as a complimentary metal-oxide-semiconductor (CMOS) design. This gate uses 2x p-type FET and 2x n-type FET. The equivlanet fault tolerant gate design is illustrated in Figure 2b where additional FETs and interconnects have been added. The interconnect topology shown is one of several variations under investigation [12]. The design is intended to bring the capabilities of stuck-off fault masking and detection of stuck-high faults.

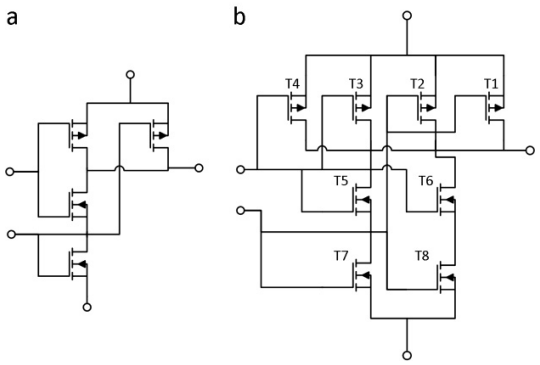

Figure 2 (a) traditional NAND gate design. (b) NAND gate design with fault discrimination and masking capability.

For simplicity it is assumed that each FET is fabricated separately i.e, that there is no overlapping of resources in the layout. For CMOS circuits this may not be the case because combined structures are commonly used. However the degree of resource overlap depends on the technology library used hence we adopt a simple model that can be later refined using a specific technology library. The predicted fault rate of the NAND gate design is shown in Figure 3 alongside other variations. The basic NAND gate is shown leftmost on the horizontal axis and each redundant design "NAND+n" contains n redundant FETs. The full quad-transistor (QT) design (NAND+12) achieves complete fault tolerance against and stuck-on and stuck-off fault [13], but with a 4x resource overhead requirement in comparison to the non-redundant design (NAND). The design presented here is represented by the NAND+4 variation with a fault rate of 12.5%.

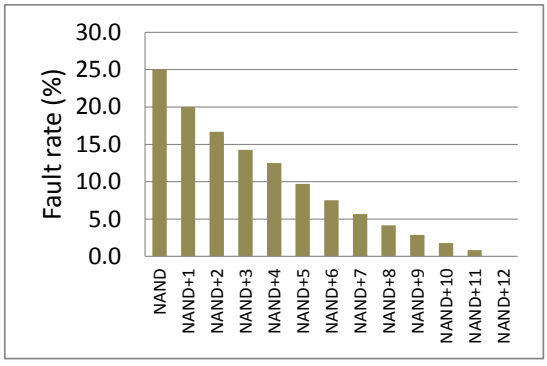

Figure 3 Fault rate versus NAND gate redundancy.

The concept of augmenting resilience with fault event intelligence has been suggested by others. For example, a radiation sensor is incorporated into an FPGA-based sub-system in [14] to create environmental awareness that enhances the basic TMR/scrubbing procedures. By comparison our strategy is implemented using an analgoue trigger flag referred to as IDDQ. This is an attractive approach because detection is confined to the electronic circuit domain rathern than relying upon external sensory input. Furthermore, active responses triggered by IDDQ events are controlled by local circuitry and hence become extremely rapid and potentially autonomous. The concept is illustrated in Figure 4, where a stuck-on fault has been asserted at FET T7. In this case IDDQ current flows when the inputs are set to '01'. When IDDQ current flows the output logic state is difficult to predict as it depends upon the analogue conditions of drain-source impedance of each FET. Therefore, the output is not considered trustworthy and should be ignored. However the ocurrence of IDDQ can be used to identify the stuck-on fault condition.

3. Test strategy

To confirm the properties of the fault tolerant design we implemented the NAND gate using discrete FETs mounted on a test PCB. Fault-tolerant designs are typically evaluated using either software simulation or else using FPGA boards in order to predict their usefulness. A discussion of the different test approaches can be found in [15]. Examples of model-based approaches are seen in [1,2,5] wherein behavioural predictions of the fault response are formed. Alternatively, hardware fault injection within FPGA boards has been carried out taking the form of random bit-flips injected into the configuration bitstream. Examples of this are seen in [16] for evaluation of FPGA fault-tolerant design techniques. The faults injected are emulated, that is, artificially inserted into the active hardware by a separate hardware controller. Some modern FPGA chips benefit from a built-in fault injection interface [17]. Finally, accelerated radiation testing constitutes the ultimate form of fault testing whereby faults are induced by the actual physics of failure mechanism involving high energy particles interacting with semi-conductor materials.

For the purposes of this study we have chosen to adopt fault emulation by hardware injection due to the high repeatability of the approach and lower cost in comparison to accelerated testing. For the fault tolerant design under consideration, hardware fault injection offers the possibility to monitor both analogue and digital domain behaviour. This is essential when observing the stuck-high behaviour for the case when rail to rail current flows. This condition is conventionally ignored in fault analysis of logic circuits (see for example [18]) because the logic output level becomes ill-defined. However our fault detection strategy relies only upon the presence of (analogue) IDDQ current flow rather than determination of the specific (digital) logic level.

Detailed information about IDDQ is not available using FPGA chips due to their architecture, hence a custom circuit implementation was created. This also created the possibility of asserting different fault conditions for each FET. The FETs used are low-power MOSFETs (types IRFD020 and IDRF9024) rated at 1 Watt, hence the circuit could also be used to demonstrate a small-scale power application. The test PCB (Figure 5) is fitted with fault injection points that allow insertion of stuck-on / stuck-off conditions for any FET. In addition are also fault injection points for gate signals that are not used in this experiment. The PCB is wired to a National Instruments PXI test system.

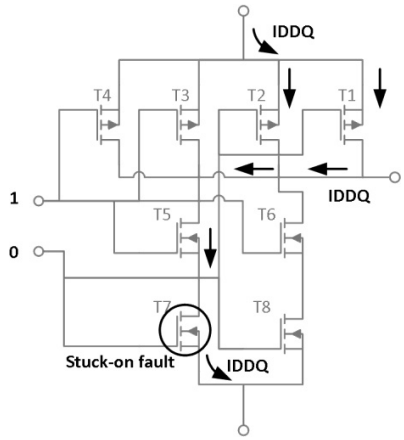

Figure 4 Example of IDDQ triggered by stuck-on fault occurring at T7.

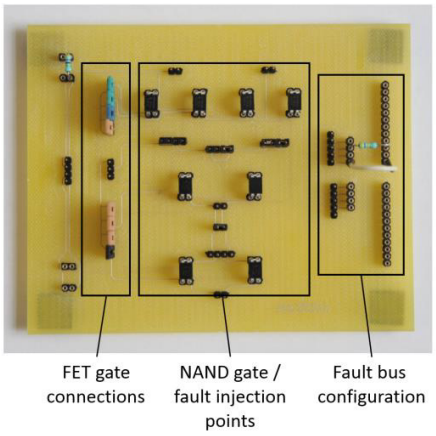

Figure 5 Photo of test PCB showing key features of NAND gate composed of FETs and configurable connection points for fault injection.

Fault injection is controlled by a fault insertion unit (FIU) type PXI-2510 capable of coordinating up to 64 fault channels across two fault buses. The basic topology is illustrated in Figure 6. To inject a fault, relays are opened/closed such that relevant FET is connected in parallel with different fault loads. The test PCB includes connection points for up to four different fault loads for each fault bus. In this experiment, the fault loads comprise a short circuit track and a 500 kΩ resistor. Gate-level testing includes digital response test and analogue IDDQ measurement. A high-speed digital I/O module was programmed to test the NAND gate response during each fault event. IDDQ was recorded using a PXI-DMM module connected in-line with the power rail.

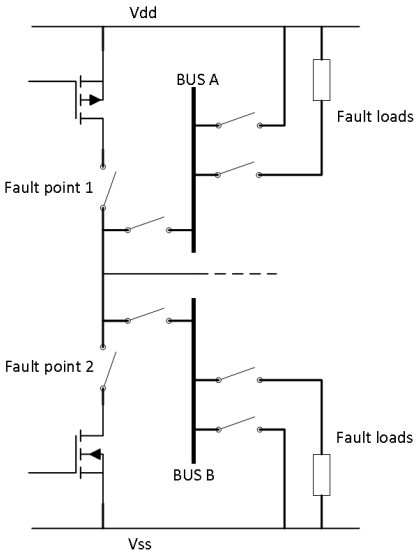

Figure 6 Fault injection schematic showing two fault points, fault buses and fault loads.

Testing is coordinated using a LabVIEW test panel that allows users control of the test configuration and output data (Figure 7). The associated virtual instrument (VI) panel controls the test sequence and data retrieval. Each test sequence comprises the following operations:

- Initialise hardware
- Load scan list(s) from file(s)
- Set fault condition
- Execute digital I/O test
- Measure IDDQ
- Add data to results matrix

Analogue/digital measurement is coordinated by a separate VI. Fault injection is controlled by a scan list, which contains a sequence of ASCII commands that initiate each fault connection. An example scan list is:

~dut0->ch0 && dut0->busA & a1->busA;
~a1->busA && a0->busA;
~a0->busA && ~busA->dut0 & ch0->dut0;

In this example fault location “dut0” is connected to fault Bus A, then fault load “a0” (stuck-on fault) and load “a1” (stuck-open) are connected. The fault location is then returned to normal operation. Many such scan list files are read and executed by the VI. Results are organised into matrices and saved in csv format for analysis.

Finally the experimental test system components are shown in Figure 8.

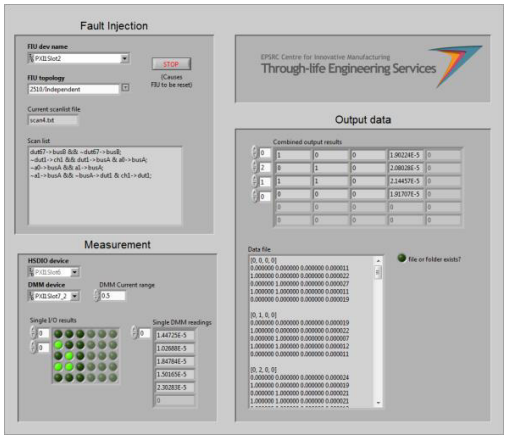

Figure 7 LabVIEW control panel for fault injection experiments.

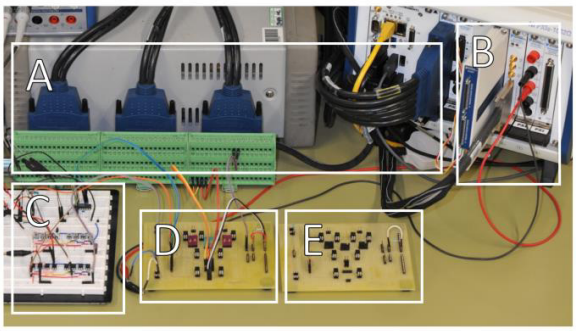

Figure 8 Photo showing experimental set up. A: fault injection unit; B: digital and analogue test electronics; C: ancillary test electronics; D: test NAND gate PCB; E: additional test PCB.

4. Results

After testing is complete the results are easily analysed using Excel macros. A key result is that all stuck-off faults were successfully masked by the NAND gate design. Stuck-on results are shown in Figure 9, where the circuit response has been recorded for a single stuck-on fault asserted at each FET T1…T8. The green indicators show where an IDDQ event has been detected and hence where stuck-on fault can be detected. The data shows that an IDDQ event occurs once for every fault location and hence the stuck-at fault is always detectable. The fault rate is therefore 25% for stuck-high faults and 12.5% considering both stuck-high/low faults.

Stuck-high fault location	Input	Output	IDDQ	Input	Output	IDDQ	Input	Output	IDDQ	Input	Output	IDDQ
T1	'00'	H	🔴	'01'	H	🔴	'10'	H	🔴	'11'	H	🟢
T2	'00'	H	🔴	'01'	H	🔴	'10'	H	🔴	'11'	H	🟢
T3	'00'	H	🔴	'01'	H	🔴	'10'	H	🔴	'11'	H	🟢
T4	'00'	H	🔴	'01'	H	🔴	'10'	H	🔴	'11'	H	🟢
T5	'00'	H	🔴	'01'	H	🔴	'10'	L	🟢	'11'	L	🔴
T6	'00'	H	🔴	'01'	H	🔴	'10'	L	🟢	'11'	L	🔴
T7	'00'	H	🔴	'01'	L	🟢	'10'	H	🔴	'11'	L	🔴
T8	'00'	H	🔴	'01'	L	🟢	'10'	H	🔴	'11'	L	🔴

Figure 9 Results from sample experimental fault injection campaign.

IDDQ occurs for different input states depending upon the fault location, hence a digital response test is required to trigger IDDQ detection. Although this is a potentially time-intensive operation there is the possibility of built-in fault localisation. For example, if IDDQ occurs for the input pattern '10' then a stuck-at high fault must be located at either T5 or T6.

Another feature is that output errors only occur during IDDQ events. Hence the circuit output could still be considered trustworthy except when IDDQ events occur and therefore the gate's output could still be used in 75% of stuck-high fault conditions and discarded whenever a IDDQ event occurs.

5. Conclusions

An experimental test bench has been used to demonstrate a novel fault tolerant design of a NAND gate. Three key features have been confirmed via fault injection and analogue measurement: 1) masking of all stuck-off faults; 2) IDDQ event for all stuck-on faults; 3) fault localisation by combing IDDQ event and logic input pattern. Further tests have shown that the NAND gate also tolerates a limited number of double stuck-off faults. In some situations current flow between VDD and VSS would risk damage to the active FETs and should be avoided. In these cases the IDDQ event could be used to trigger an auto-switchover mechanism whereby a standby NAND gate takes the place of the faulty gate. An important benefit here would be that switch-over becomes selective and only occurs upon detection of a damaging stuck-high fault condition rather than stuck-off faults or dormant stuck-high faults.

The demonstrated test bench implementation is extensible to 64 fault locations, with further extension possible using a multiplexing switch unit. Complex fault patterns are easily programmed via scan list files. Hence the approach is capable of scaling to evaluation of fault discrimination/masking in logic units composed of multiple NAND gates such as basic arithmetic logic units. By generalising the fault injection patterns to include multiple simultaneous faults (along with IDDQ measurement) important fault rates data may then be measured and accumulated for reliability calculations according to the procedure described in [2]. This will lead to a better understanding of the resource/performance trade-offs incurred in the design of fault tolerant electronics.

Acknowledgements

This work was carried out with the support of the EPSRC Innovative Centre for Through-life Engineering Services [EP/I033246/1].

References

[1] Breuer MA, Gupta SK, Mak TM. Defect and error tolerance in the presence of massive numbers of defects. IEEE Des Test Comput. 2004;21(3):216–27.
[2] El-Maleh AH, Al-Hashimi BM, Melouki A, Khan F. Defect-tolerant n2-transistor structure for reliable nanoelectronic designs. IET Comput Digit Tech. 2009;3(6):570–80.
[3] Von Neumann J. Probabilistic logics and the synthesis of reliable organisms from unreliable components. Autom Stud. 1956;34:43–98.
[4] Henkel J, Bauer L, Dutt N, Gupta P, Nassif S, Shafique M, et al. Reliable On-chip Systems in the Nano-era: Lessons Learnt and Future Trends. Proceedings of the 50th Annual Design Automation Conference. New York, NY, USA: ACM; 2013;Article No. 99. Available from: http://doi.acm.org/10.1145/2463209.2488857
[5] Jensen PA. Quadded NOR Logic. IEEE Trans Reliab. 1963 Sep;R-12(3):22–31.
[6] Kastensmidt FL, Reis R, Carro L. Fault-Tolerance Techniques for SRAM-Based FPGAs. Springer; 2007.
[7] Parris MG, Sharma CA, Demara RF. Progress in Autonomous Fault Recovery of Field Programmable Gate Arrays. ACM Comput Surv. 2011;43(4):31:1–31:30.
[8] Cheatham JA, Emmert JM, Baumgart S. A survey of fault tolerant methodologies for FPGAs. ACM Trans Autom Electron Syst. 2006;11(2):501–33.
[9] Sandi Habnic. Functional triple modular redundancy (FTMR) [Internet]. Gaisler Research; 2002. Available from: http://www.gaisler.com/doc/fpga_003_01-0-2.pdf
[10] Abramovici M, Breuer MA, Friedman EG. Digital Systems Testing and Testable Design. Wiley-IEEE Press; 1990.
[11] Maxion RA, Siewiorek DP, Elkind SA. Techniques and Architectures for Fault-Tolerant Computing. Annu Rev Comput Sci. 1987;2(1):469–520.
[12] Schiefer P, McWilliam R, Purvis A. Fault Tolerant Quadded Logic Cell Structure with Built-in Adaptive Time Redundancy. Procedia CIRP. 2014;22:127–31.
[13] Han J, Leung E, Liu L, Lombardi F. A Fault-Tolerant Technique Using Quadded Logic and Quadded Transistors. IEEE Trans Very Large Scale Integr VLSI Syst. 2015;23(8):1562-1566.
[14] Hane JS, LaMeres BJ, Kaiser T, Weber R, Buerkle T. Increasing Radiation Tolerance of Field-Programmable-Gate-Array-Based Computers Through Redundancy and Environmental Awareness. J Aerosp Inf Syst. 2014;11(2):68–81.
[15] Battezzati N, Sterpone L, Violante M. Reconfigurable field programmable gate arrays for mission-critical applications. Springer Science & Business Media; 2010.
[16] Sterpone L. Electronics System Design Techniques for Safety Critical Applications. 1st ed. Springer; 2008.
[17] Xilinx Application Note. Demonstration of soft error mitigation IP and partial reconfiguration capability on monolithic devices [Internet]. 2015 [cited 2015 Aug 3]. Available from: http://www.xilinx.com/support/documentation/application_notes/xapp1261-demo-sem-pr.pdf
[18] Wadsack RL. Fault Modeling and Logic Simulation of CMOS and M08 Integrated Circuits. Bell Syst Tech J. 1978.